Bab I
Pendahuluan

1.1 Latar Belakang

Perkembangan dari analisis pola saat ini tidak lepas dari peran graf. Data yang memiliki lebih dari banyak domain direpresentasikan ke dalam graf untuk mempermudah proses manipulasi data [5]. Senyawa kimia, peta geografik, jaringan komputer, dan *database* adalah beberapa contoh data yang telah direpresentasikan ke dalam bentuk graf. Graf merupakan kumpulan dari *vertex* dan *edge*. *Vertex* digambarkan sebagai sebuah titik atau objek. Sedangkan *edge* digambarkan sebagai sebuah sisi yang menghubungkan suatu *vertex* dengan *vertex* lain.

Dari sudut pandang analisis pola pada graf, salah satu masalah yang penting adalah pencocokan *subgraph* atau pencocokan *graph pattern* pada graf untuk membandingkan keduanya[2]. Pertumbuhan keragaman dan ukuran data yang semakin mengacu pada data graf membuat model data, *query language* dan sistem *database* yang ada saat ini tidak mendukung pemodelan dan manajemen data tersebut[5]. Berbagai penelitian telah dilakukan untuk mengatasi masalah pencarian *graph pattern* pada graf yang besar dengan jumlah *vertex* diatas 100[2]. Kompleksitas komputasi algoritma menjadi pertimbangan dalam pencarian *graph pattern* pada graf. Pada graf yang besar memerlukan kompleksitas komputasi yang rendah untuk mempercepat proses pencarian *graph pattern*. Salah satu teknik yang digunakan adalah dengan melakukan pemangkasan *vertex* dan *edge* yang tidak memiliki keterkaitan dengan pola yang dicari. Pencarian juga tidak sebatas pada pencarian informasi *vertex* dan *edge* saja. Informasi struktural dari *vertex* dan *edge* pada objek yang dicari sangat diperlukan agar informasi yang didapatkan tidak ada yang hilang[5]. Oleh karena itu, diperlukan suatu metode pencarian *graph pattern* dalam graf yang memiliki kemampuan untuk memberikan hasil berupa graf atau kumpulan dari beberapa graf.

Graph Query Language(GraphQL) merupakan salah satu metode yang dapat digunakan untuk mencari *graph pattern* dalam suatu graf. GraphQL dapat digunakan pada graf yang berukuran besar. Teknik yang digunakan dalam pencarian pola adalah melakukan pemangkasan terhadap *vertex* dan *edge* pada graf, yang tidak memiliki keterkaitan dengan pola yang terdapat pada *graph pattern.* Kelebihan dari pemangkasan yang dilakukan oleh algoritma ini adalah pemangkasan yang dilakukan dua kali yaitu pemangkasan lokal dan pemangkasan global[7]. Hasil yang didapatkan dari pencarian dengan algoritma GraphQL adalah kumpulan dari beberapa graf.

1.2 Perumusan Masalah

Permasalahan yang diangkat dalam Tugas Akhir ini adalah:

1. Bagaimana menerapkan algoritma GraphQL (*Graph Query language*) dalam pencocokan *graph database* pada *dataset*?
2. Bagaimana performansi *running time* dan *reduction ratio* algoritma GraphQL (*Graph Query Language*)?

1.3 Tujuan

Tujuan yang ingin dicapai dari Tugas Akhir ini adalah:

1. Melakukan penerapan algoritma GraphQL(*Graph Query Language*) dalam pencocokan *graph database* pada *dataset*
2. Melakukan analisis untuk mengetahui performansi *running time* dan *reduction ratio* algoritma GraphQL (*Graph Query Language)*

1.4 Batasan Masalah

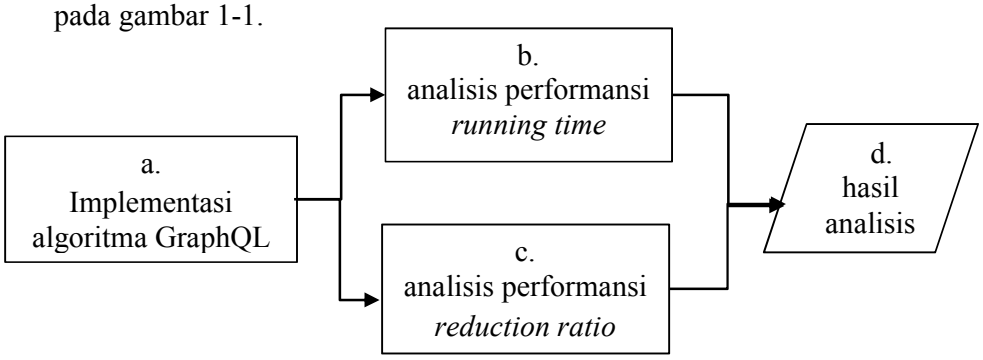
Adapun batasan masalah dalam Tugas Akhir ini adalah:

1. Data *graph database* sebagai data uji lebih dari satu label
2. Graf yang dibangun berupa graf tidak berarah yang sederhana
3. *Graph pattern* yang digunakan berupa lintasan dan *clique*
4. Menggunakan bahasa pemrograman java.

1.5 Metodologi Penyelesaian Masalah

Metodologi penyelesesaian masalah terbagi dalam beberapa tahapan, yaitu:

1. Studi Literatur
   Melakukan pengumpulan dan mempelajari data, informasi serta berbagai referensi yang berkaitan dengan pencocokan graf serta algoritma GraphQL sebagai konsep pengerjaan tugas akhir.
2. Pengumpulan dan Pengolahan Data
   Mencari studi kasus berupa *dataset graph database* yang digunakan untuk mengimplementasikan algoritma GraphQL dalam pencocokan *graph pattern* pada *graph database.*
3. Desain Sistem
   Melakukan perancangan serta pendefinisian masalah dan solusi yang diterapkan ke dalam implementasi sistem.
4. Implementasi
   Melakukan implementasi algoritma GraphQL untuk melakukan pencocokan *graph pattern* pada *graph database*.
5. Kesimpulan dan Laporan Hasil Analisis
   Dalam pengambilan kesimpulan, yang dianalisis adalah data yang didapat dari pengujian performansi *running time* dan *reduction ratio* dari algoritma GraphQL. Pengambilan kesimpulan menggunakan langkah-langkah seperti pada gambar 1-1.



Gambar 1-1: Alur Pengambilan Keputusan

Pada gambar 1-1, implementasi terhadap algoritma GraphQL pada *graph database* dilakukan terlebih dahulu. Dalam proses implementasi tersebut dilakukan pengamatan terhadap performansi performansi *reduction ratio* pada tahapan pemangkasan dan *running time* setiap tahapan algoritma GraphQL. Analisis *running time* untuk mengetahui lama waktu setiap proses dari algoritma GraphQL dan analisis *reduction ratio* untuk mengetahui keefektifan setiap tahap pemangkasan algoritma GraphQL.

6. Dokumentasi
   Dari setiap tahapan yang dilakukan, dibuatkan dokumentasi agar setiap kegiatan dapat dipertanggungjawabkan secara jelas.